Лабораторная работа 13.

Черновик 0.5

Цель работы – приобрести навыки работы с готовыми "компонентами", закрепить навыки работы со списочными структурами и динамической памятью.

Студенты должны получить и закрепить на практике следующие знания и умения:

1. Использование готовых компонент (реализация списка).
2. Использование различных списочных структур.
3. Грамотная работа с ресурсами (динамическая память, файлы).

Общее задание

1. Исходный код лабораторной работы располагается в ветке lab_13. В этой ветке создается папка lab_13, в которой располагается решение задачи.
2. Исходный код должен соответствовать правилам оформления исходного кода.
3. Решение задачи оформляется как многофайловый проект. Сборка проекта выполняется с помощью утилиты make. В сценарии сборки должны присутствовать следующие цели: *app.exe* для сборки приложения, *test.exe* для сборки модульных тестов, *ftest* для запуска функциональных тестов.
4. В модульных тестах необходимо добиться 100% покрытия кода. Исключением являются строки с использованием функций выделения памяти (ситуация, когда память не удалось выделить).
5. Для функциональных тестов студентом подготавливаются тестовые данные, которые демонстрируют правильность работы программы. Входные данные должны располагаться в файлах in_z.txt, выходные out_z.txt, где z – номер тестового случая. Тестовые данные готовятся и помещаются под версионный контроль еще до того, как появится реализация задачи. Запуск программы с различными тестовыми данными необходимо выполнить в самом сценарии сборки или в sh-скриптах (аналог cmd-файлов). Для проверки совпадения ожидаемого и фактического результатов необходимо использовать команду diff. Запуск функциональных тестов должен выполняться с помощью цели *ftest*.

2. Индивидуальное задание

В текстовом файле хранится структурированная информация о некоторой предметной области. Информация о каждом элементе этой области располагается на отдельной строке, значения полей разделены точкой с запятой (справа и слева от нее могут быть пробелы).

Например, тестовый файл хранит информацию о книгах. Каждая книга в этом файле описывается названием, автором, годом издания, издательством, местом издания и т.п. Информация об авторе содержит фамилию и инициалы (именно в таком порядке). Будем предполагать, что у книги только один автор.

Необходимо реализовать простую базу данных.

Информация из текстового файла загружается в оперативную память (она хранится в классическом списке). Для работы с этой информацией предусмотрены следующие функции

- добавление элемента;

- удаление элемента;
- изменение сведений об элементе;
- сохранение информации на диск.

Для удобства работы предусмотрены
- индексы, с помощью которых исходную информацию можно получить в упорядоченном виде (например, книги можно упорядочить по автору, по автору и названию, по году издания и издательству);
- фильтры, содержащие сведения об элементах удовлетворяющих заданным критериям (например, только книги указанного автора, книги в названии которых есть указанное слово).

Количество индексов - 3, фильтров - 2. Они реализуются с помощью списков из ядра Linux.

Взаимодействие с пользователем осуществляется с помощью меню. Например,
1. Load from file
2. Add record
3. Del record by name

и т.д.

Классический список содержащий информацию из предметной области должен выглядеть примерно так

```
#include "list.h"

#define MAX_SORT_INDX    3
#define MAX_FILTER_INDX    2

struct book
{
    char *title;
    char *author;
    unsigned short year;
    char *publ_house;
    // ...

    struct list_head sort[MAX_SORT_INDX];
    struct list_head filter[MAX_FILTER_INDX];

    struct book *next;
};
```

Обратите внимание на то, что где-то нужно хранить указатели на "первый" элемент в списках индексов и фильтров.

Замечание 1.
Настоятельно рекомендуется выбрать собственную предметную область, например, песни/диски/фильмы/картины и т.д.

Замечание 2.

Индексы с помощью списков никто не реализует, потому что при добавление нового элемента, чтобы перестроить индекс список придется просматривать целиком. Обычно индексы реализуются деревьями, но мы лишь мельком рассмотрели деревья.